UECM1703 Introduction of Scientific Computing Marking Guide

TOPIC 6 Practical

UNIVERSITI TUNKU ABDUL RAHMAN

Faculty:	FES	Unit Code:	UECM1703
Course:	AM &FM	Unit Title:	Introduction To Scientific Computing
Year:	1&2	Lecturer:	Dr Yong Chin Khian
Session:	Oct 2023		

Q1. A local Bank wants to estimate the number of foreclosures per 1000 houses sold. Bank officials think that the number of foreclosures is related to the size of the down payment made by house buyers. The following table contains foreclosure and down payment data.

Down Payment Size (% of purchase price)	Number of Foreclosures per 1000 houses
10	37
20	15
14	27
12	29
18	12
16	25

You fit the above data to $Y = \beta_0 + \beta_1 X + \epsilon$, where Y is the number of foreclosures per 1000 houses sold, and x is the size of the down payment made by house buyers. Use the stat.models module to answer the following questions:

(a)State the estmated regression function.

Ans.

$\hat{Y} = 59.095238095238074 - 2.3285714285714256X$

```
import numpy as np
import statsmodels.api as sm
y = ([37,15,27,29,12,25])
n = len(y)
x0 = np.ones(n).reshape(-1,1)
x1 = np.array([10,20,14,12,18,16]).reshape(-1,1)
x = np.hstack((x0, x1))
x,y = np.array(x), np.array(y)
model = sm.OLS(y, x)
results = model.fit()
print(results.summary())
b = results.params
print("b = ", b)
```

```
Output:
b =  [59.095238095238074 -2.3285714285714256]
```

(b)Obtain a prediction for a new observation y_h when $x_h = 17.0$.

Ans.

$y_h = 59.0952 + -2.3286(17.0) = 19.5095$

```
Python code:
xh = np.array([1,17.0])
yh = results.predict(xh)
print("yh = ",yh)
Output:
yh = [[19.50952381]]
```

(c)Compute the R^2.

Ans.

$R^2 = 0.8850924435067532$

```
print("R-squared = ", results.rsquared)
Output:
R-squared =  0.8850924435067532
```

(d)Compute the adjusted R^2.

Ans.

Adjusted $R^2 = 0.8563655543834414$

```
print("Adj. R-squared = ", results.rsquared_adj)
Output:
Adj. R-squared =  0.8563655543834414
```

(e)Compute the AIC.

Ans.

AIC $= 33.66135165861845$

```
print("AIC = ", results.aic)
Output:
```

```
AIC =  33.66135165861845
```

(f)Compute the BIC.

Ans.

BIC = 33.24487059707456

```
print("BIC = ", results.bic)
Output:
BIC =  33.24487059707456
```

(g)Compute the test statistic for testing $H_0 : \beta_1 = 0$ versus $H_1 : \beta_1 \neq 0$

Ans.

t = -5.550728908767898

```
print("t = ", results.tvalues[1])
Output:
t =  -5.550728908767898
```

(h)Compute the p value corresponding to the test above.

Ans.

$pvalue$ = 0.005154339843251439

```
print("p value = ", results.pvalues[1])
Output:
p value =  0.005154339843251439
```

Q2. Consider the data shown below:

y	x	y	x
30037	16	13285	13
4763	10	6909	11
6909	11	30037	16
3161	9	662	6
17769	14	6909	11
662	6	6909	11
38137	17	23302	15
4763	10	3161	9

(a) Write down the python codes to fit the data to the polynomial regression model of degree 4.

Ans.

```
import numpy as np
import matplotlib.pyplot as plt
x = [16,10,11,9,14,6,17,10,13,11,16,6,11,11,15,9]
y = [30037,4763,6909,3161,17769,662,38137,4763,
13285,6909,30037,662,6909,6909,23302,3161]
#fit polynomial models up to degree 4
model4 = np.poly1d(np.polyfit(x, y, 4))
```

(b) Continue using the codes from part (a), write down the python codes to created a scatter plot of the data and add the fitted polynomial line to the scatterplot.

Ans.

```
#create scatterplot
import matplotlib.pyplot as plt
polyline = np.linspace(5, 20, 50)
plt.scatter(x, y)
plt.title("y vs x")
plt.xlabel("x")
plt.ylabel("y")
#add fitted polynomial lines to scatterplot
plt.plot(polyline, model4(polyline), color='blue')
plt.show()
```

Q3. You fit $Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \epsilon$ with independent normal error terms to data in data file dfQ09c.csv. Use the stat.models package in Python to answer the following questions.

(a) State the estimated regression function.

Ans.
$\hat{Y} = 63.9136 + 1.1551X_1 + 0.3039X_2 + 1.2871X_3 = 2.2868X_4$

(b) Obtain a prediction for a new observation Y_h when $x_{h_1} = 5.9$, $x_{h_2} = 30.8$, $x_{h_3} = 10.2$ and $x_{h_4} = 19.0$.

Ans.
$\hat{Y} = 63.9136+1.1551(5.9)+0.3039(30.8)+1.2871(10.2)+2.2868(19.0) = 136.6662$

(c) Write down the python codes and the correspoding outputs that you obtained for answers in part (a) and part (b).

Ans.

Python code:

```
import pandas as pd
import numpy as np
import statsmodels.api as sm
import matplotlib.pyplot as plt
df = pd.read_csv('dfQ09c.csv')
y = df.y
n = len(y)
x0 = np.ones(n).reshape(-1,1)
x1 = np.array(df.x1).reshape(-1,1)
x2 = np.array(df.x2).reshape(-1,1)
x3 = np.array(df.x3).reshape(-1,1)
x4 = np.array(df.x4).reshape(-1,1)
x = np.hstack((x0,x1,x2,x3,x4))
x, y = np.array(x), np.array(y)
model = sm.OLS(y, x)
results = model.fit()
Beta = results.params
print("beta = ",Beta)
x_new = np.array([1,5.9,30.8,10.2,19.0]).reshape(-1,5)
print("x_new=",x_new)
y_new = results.predict(x_new)
print("y_new=",y_new)
#print(results.summary())
Output:
Beta hat =  [63.9136 1.1551 0.3039 1.2871 1.2871]
x_new= [[ 1.   5.9 30.8 10.2 19.0]]
y_new= [136.6662]
```

Q4. Consider the data below:

y	x_1	x_2	x_3	x_4
265.1	7.9	31.6	15.4	30.2
179.8	4.6	29.4	8.7	16.8
182.9	4.7	29.5	8.9	17.3
162.5	3.9	28.9	7.3	14.1
236.4	6.8	30.9	13.1	25.7
115.4	2.0	27.7	3.6	6.7
270.0	8.1	31.7	15.7	31.0
174.1	4.4	29.2	8.2	15.9
215.5	6.0	30.3	11.5	22.4
193.8	5.1	29.8	9.8	19.0

Assuming that regression model $y = \beta_0+\beta_1x_1+\beta_2x_2+\beta_3x_3+\beta_4x_4+\epsilon$ is appropriate.

(a) Find the test statistic for testing $H_0 : \beta_4 = 0$.

Ans.

T = 26.973233950590817

```
import numpy as np
import statsmodels.api as sm
y = [265.1, 179.8, 182.9, 162.5, 236.4,
115.4, 270.0, 174.1, 215.5,193.8]
n = len(y)
x0 = np.ones(n).reshape(-1,1)
x1 = np.array([7.9, 4.6, 4.7, 3.9, 6.8,
2.0, 8.1, 4.4, 6.0, 5.1]).reshape(-1,1)
x2 = np.array([31.6, 29.4, 29.5, 28.9, 30.9,
27.7, 31.7, 29.2, 30.3, 29.8]).reshape(-1,1)
x3 = np.array([15.4, 8.7, 8.9, 7.3, 13.1,
3.6, 15.7, 8.2, 11.5, 9.8]).reshape(-1,1)
x4 = np.array([30.2, 16.8, 17.3, 14.1, 25.7,
6.7, 31.0, 15.9, 22.4, 19.0]).reshape(-1,1)
x = np.hstack((x0, x1,x2,x3,x4))
x, y = np.array(x), np.array(y)
model = sm.OLS(y, x)
results = model.fit()
T = results.tvalues[4]
print("T = ", T)
Output:
T = 26.973233950590817
```

(b) Find the p-value for testing $H_0 : \beta_4 = 0$.

Ans.

P-Value = 1.3099790361287864e-06

```
pv = results.pvalues[4]
print("pvalue", pv)
Output: 1.3099790361287864e-06
```

(c) Find the test statistic for testing $H_0 : \beta_1 = \beta_2 = \beta_3 = \beta_4 = 0$.

Ans.

F = 3861045.646045324

```
F = results.fvalue
print("F = ", F)
Output:
F = 3861045.646045324
```

(d) Find the p-value for testing $H_0 : \beta_1 = \beta_2 = \beta_3 = \beta_4 = 0$.

Ans.

P-Value = 2.0872779488715573e-16

```
pv2 = results.f_pvalue
print("pvalue = ", pv2)
Output:
pvalue = 2.0872779488715573e-16
```

Q5. Consider the data shown below:

y	x	y	x
1062294	32	398299	25
164352	20	239822	22
164352	20	936299	31
86354	17	21832	12
624752	28	199420	21
10659	10	199420	21
1200597	33	718245	29
134126	19	86354	17

Fit the polynomial regression models up to order 4, then answer the following questions:

(a)Fill in the table below:

Polynomial model	Adjusted R^2	AIC	BIC
First order			
Second order			
Third order			
Fourth order			

(b)Determine a model that best fit the data based on the adjusted R^2.

(c)Write down the equation of the best fitted curve that you have selected from part (b).

(d)Created a scatter plot of the data and add the fitted polynomial line to the scatterplot.

Ans.

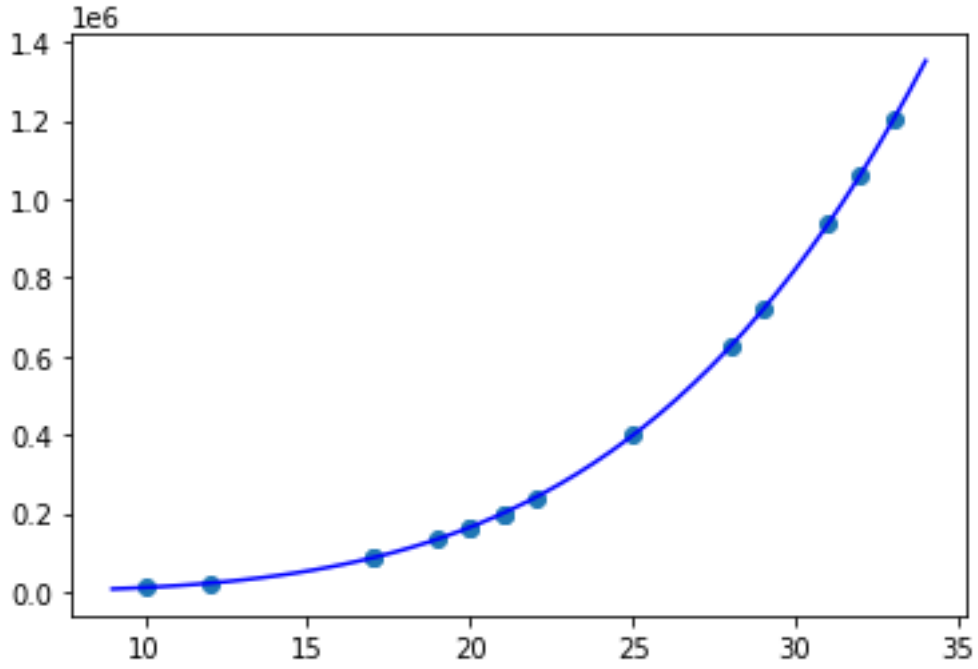
(a)

Polynomial model	Adjusted R^2	AIC	BIC
First order	0.861	427.8	429.3
Second order	0.996	370.8	373.1
Third order	1.0	285.8	288.9
Fourth order	1.0	2.753	6.616

(b)Since the fourth-order polynomial has the largest adjusted R^2 thus, model with polynomial fourth degree should be selected.

(c)$y = -6.3931 + 1.61x + 0.1671x^2 + 0.7331x^3 + 0.9900x^4$

(d)




```
import numpy as np
from sklearn.preprocessing import PolynomialFeatures
```

```
import statsmodels.api as sm
import matplotlib.pyplot as plt
y = np.array([1062294,164352,164352,86354,624752,10659,1200597,134126,
              398299,239822,936299,21832,199420,199420,718245,86354])
print(y)
x = np.array([32,20,20,17,28,10,33,19,
              25,22,31,12,21,21,29,17]]).reshape((-1,1))
x_ = [32,20,20,17,28,10,33,19,
              25,22,31,12,21,21,29,17]
x1st = PolynomialFeatures(degree=1, include_bias=True).fit_transform(x)
x2nd = PolynomialFeatures(degree=2, include_bias=True).fit_transform(x)
x3rd = PolynomialFeatures(degree=3, include_bias=True).fit_transform(x)
x4th = PolynomialFeatures(degree=4, include_bias=True).fit_transform(x)
model1 = sm.OLS(y, x1st)
results1 = model1.fit()
model2 = sm.OLS(y, x2nd)
results2 = model2.fit()
model3 = sm.OLS(y, x3rd)
results3 = model3.fit()
model4 = sm.OLS(y, x4th)
results4 = model4.fit()
print("Adj. R-squared model1 = ", results1.rsquared_adj)
print("AIC model1 = ", results1.aic)
print("BIC model1 = ", results1.bic)
print("Adj. R-squared model2 = ", results2.rsquared_adj)
print("AIC model2 = ", results2.aic)
print("BIC model2 = ", results2.bic)
print("Adj. R-squared model3 = ", results3.rsquared_adj)
print("AIC model3 = ", results3.aic)
print("BIC model3 = ", results3.bic)
print("Adj. R-squared model4 = ", results4.rsquared_adj)
print("AIC model4 = ", results4.aic)
print("BIC model4 = ", results4.bic)
print("Beta hat = ", results4.params)
#create scatterplot
polyline = np.linspace(9.0, 34.0, 50)
plt.scatter(x, y)
model = np.poly1d(np.polyfit(x_, y, 4))
plt.plot(polyline, model(polyline), color='blue')
plt.show()
Output:
Adj. R-squared model1 =  0.861
AIC model1 =  427.8
BIC model1 =  429.3
Adj. R-squared model2 =  0.996
```

```
AIC model2 =  370.8
BIC model2 =  373.1
Adj. R-squared model3 =  1.0
AIC model3 =  285.8
BIC model3 =  288.9
Adj. R-squared model4 =  1.0
AIC model4 =  2.753
BIC model4 =  6.616
Beta hat =  [-6.3931 1.6080 0.1671 0.7331 0.9900]
```

Q6. A soft drink manufacturer use five agents (1,2,3,4,5) to handle premium distributions for its various products. The marketing director desired to study the timeliness with which the premiums are distributed. Six transactions for each agent were selected at random, and the time lapse (in days) for handling each transaction was determined. The results follow.

Agent	Time lapse (in days)					
1	23	26	29	29	27	27
2	22	23	20	27	20	24
3	20	21	29	30	10	23
4	27	34	27	28	23	28
5	31	22	28	27	32	29

Consider the model $y_{ij} = \mu_i + \epsilon_{ij}, i = 1, 2, 3, 4, 5; j = 1, 2, \ldots, 6$, where

- y_{ij} is the time lapse of the j^{th} transaction with i^{th} agent.
- μ_i is the mean time lapse of the i^{th} agent..
- $\epsilon_{ij} \sim N(0, \sigma^2)$

The hypotheses to test the equalities of means are $H_0 : \mu_1 = \mu_2 = \mu_3 = \mu_4 = \mu_5$ versus H_1: at least one population mean is different from the rest. Use the f_oneway package to compute the test statistic and the p value.

Ans.

F = 3.1461594466690936 p value = 0.0317252823789221

```
from scipy.stats import f_oneway
y1 = [23,26,29,29, 27]
y2 = [22,23,20,27, 20]
y3 = [20,21,29,30, 10]
y4 = [27,34,27,28, 23]
ANV = f_oneway(y1,y2,y3,y4)
```

```
print("ANOVA = ", ANV)
print("F = ", ANV[0])
print("p value = ", ANV[1])
Output:
ANOVA = F_onewayResult(statistic=3.1461594466690936, pvalue=0.0317252823789221
F = 3.1461594466690936
p value = 0.0317252823789221
```